CLAIMS

What is claimed is:

1. A method for dynamic implementation of a Java™ Metadata Interface (JMI) to a metamodel, the method comprising:

receiving a JMI implementation request, said request associated with a metamodel, said metamodel comprising at least one package, said at least one package comprising at least one class, said at least one class comprising at least one attribute, reference or operation;

implementing a package proxy JMI interface when said request comprises a package proxy request;

implementing a class proxy JMI interface when said request comprises a class proxy request; and

implementing a class instance JMI interface when said request comprises a class instance request.

2. The method of claim 1 wherein said implementing a package proxy JMI interface comprises:

generating bytecode for a class that implements said package proxy JMI interface;

creating a new instance of said class; and

returning said instance.

3. The method of claim 2 wherein said generating further comprises:

receiving a metamodel package;

receiving a package proxy interface method associated with said metamodel package;

determining a class name based upon said interface method;

searching said metamodel package for a class corresponding to said class name; and

producing an implementation of said interface method that returns the proxy for said class when said class name is found in said package.

4. The method of claim 3 wherein said implementation of said interface method calls a handler method of the superclass of said class, passing said class name as an argument and returning the proxy for said class.

5. The method of claim 1 wherein said implementing a class proxy JMI interface comprises:

generating bytecode for a class that implements said class proxy JMI interface;

creating a new instance of said class; and

returning said instance.

6. The method of claim 5 wherein said generating further comprises:

receiving a metamodel class;

receiving a class proxy interface method associated with said metamodel class;

producing a first implementation of said interface method that creates a new instance of said class when said method is parameterless; and

producing a second implementation of said interface method that creates a new instance of said class and sets the attributes passed as arguments to said interface method when said interface method includes at least one parameter.

7. The method of claim 6 wherein said first implementation calls a handler method of the superclass of said class, passing said class name as an argument and returning a new instance of said class.

8. The method of claim 6 wherein said second implementation calls a handler method of the superclass of said class, passing said class name, attributes and attribute values as arguments and returning a new instance of said class.

9. The method of claim 1 wherein said implementing a class instance JMI interface comprises:

generating bytecode for a class that implements said class instance JMI interface;

creating a new instance of said class; and

returning said instance.

10. The method of claim 9 wherein said generating further comprises:

receiving a metamodel class;

receiving a class instance interface method associated with said metamodel class, said interface method having an interface method name;

producing a first implementation of said interface method that sets the value of an attribute when said interface method name includes a first prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a second implementation of said interface method that sets the value of a reference when said interface method name includes a first prefix and when the reference associated with said interface method is found in said metamodel class;

producing a third implementation of said interface method that gets the value of an attribute when said interface method name includes a second prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a fourth implementation of said interface method that gets the value of a reference when said interface method name includes a second prefix and when the reference associated with said interface method is found in said metamodel class; and

producing a fifth implementation of said interface method that executes an operation when said interface method has the same name as said operation.

11. The method of claim 10 wherein

said first prefix is "set"; and

said second prefix is "get".

12. The method of claim 10 wherein said producing a first implementation further comprises:

receiving an attribute name and an attribute value; and

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name and attribute value as arguments.

13. The method of claim 10 wherein said producing a second implementation further comprises:

receiving a reference name and an reference value; and

producing an implementation that calls a handler method of the superclass of said class, passing said reference name and reference value as arguments.

14. The method of claim 10 wherein said producing a third implementation further comprises:

receiving an attribute name;

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name as an argument and returning the attribute value associated with said attribute name; and

returning said attribute value.

15. The method of claim 10 wherein said producing a fourth implementation further comprises:

receiving a reference name;

producing an implementation that calls a handler method of the superclass of said class, passing said reference name as an argument and returning the reference value associated with said reference name; and

returning said reference value.

16. The method of claim 10 wherein said producing a fifth implementation further comprises:

receiving an operation name and any associated arguments;

producing an implementation that calls a handler method of the superclass of said class, passing said operation name and said associated arguments as arguments and returning an operation return value; and

returning said operation return value.

17. A method for dynamic implementation of a Java™ Metadata Interface (JMI), the method comprising:

receiving a JMI implementation request, said request associated with a metamodel, said metamodel comprising at least one package, said at least one package comprising at least one class, said at least one class comprising at least one attribute, reference or operation;

implementing a JMI interface when said JMI interface is unimplemented; and

executing a stored JMI interface implementation when said JMI interface is implemented.

18. The method of claim 17 wherein

said implementing further comprises:

implementing a package proxy JMI interface when said request comprises a package proxy request and when said package proxy JMI interface is unimplemented;

implementing a class proxy JMI interface when said request comprises a class proxy request and when said class proxy JMI interface is unimplemented; and

implementing a class instance JMI interface when said request comprises a class instance request and when said class instance JMI interface is unimplemented; and

said executing further comprises:

executing a stored a package proxy JMI interface implementation when said request comprises a package proxy request and when said package proxy JMI interface is implemented;

executing a stored class proxy JMI interface when said request comprises a class proxy request and when said class proxy JMI interface is implemented; and

executing a stored class instance JMI interface when said request comprises a class instance request and when said class instance JMI interface is implemented.

19. The method of claim 18 wherein said implementing a package proxy JMI interface comprises:

generating bytecode for a class that implements said package proxy JMI interface;

creating a new instance of said class; and

returning said instance.

20. The method of claim 19 wherein said generating further comprises:

receiving a metamodel package;

receiving a package proxy interface method associated with said metamodel package;

determining a class name based upon said interface method;

searching said metamodel package for a class corresponding to said class name; and

producing an implementation of said interface method that returns the proxy for said class when said class name is found in said package.

21. The method of claim 20 wherein said implementation of said interface method calls a handler method of the superclass of said class, passing said class name as an argument and returning the proxy for said class.

22. The method of claim 18 wherein said implementing a class proxy JMI interface comprises:

generating bytecode for a class that implements said class proxy JMI interface;

creating a new instance of said class; and

returning said instance.

23. The method of claim 22 wherein said generating further comprises:

receiving a metamodel class;

receiving a class proxy interface method associated with said metamodel class;

producing a first implementation of said interface method that creates a new instance of said class when said method is parameterless; and

producing a second implementation of said interface method that creates a new instance of said class and sets the attributes passed as arguments to said interface method when said interface method includes at least one parameter.

24. The method of claim 23 wherein said first implementation calls a handler method of the superclass of said class, passing said class name as an argument and returning a new instance of said class.

25. The method of claim 23 wherein said second implementation calls a handler method of the superclass of said class, passing said class name, attributes and attribute values as arguments and returning a new instance of said class.

26. The method of claim 18 wherein said implementing a class instance JMI interface comprises:

generating bytecode for a class that implements said class instance JMI interface;

creating a new instance of said class; and

returning said instance.

27. The method of claim 26 wherein said generating further comprises:

receiving a metamodel class;

receiving a class instance interface method associated with said metamodel class, said interface method having an interface method name;

producing a first implementation of said interface method that sets the value of an attribute when said interface method name includes a first prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a second implementation of said interface method that sets the value of a reference when said interface method name includes a first prefix and when the reference associated with said interface method is found in said metamodel class;

producing a third implementation of said interface method that gets the value of an attribute when said interface method name includes a second prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a fourth implementation of said interface method that gets the value of a reference when said interface method name includes a second prefix and when the reference associated with said interface method is found in said metamodel class; and

producing a fifth implementation of said interface method that executes an operation when said interface method has the same name as said operation.

28. The method of claim 27 wherein

said first prefix is "set"; and

said second prefix is "get".

29. The method of claim 27 wherein said producing a first implementation further comprises:

receiving an attribute name and an attribute value; and

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name and attribute value as arguments.

30. The method of claim 27 wherein said producing a second implementation further comprises:

receiving a reference name and an reference value; and

producing an implementation that calls a handler method of the superclass of said class, passing said reference name and reference value as arguments.

31. The method of claim 27 wherein said producing a third implementation further comprises:

receiving an attribute name;

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name as an argument and returning the attribute value associated with said attribute name; and

returning said attribute value.

32. The method of claim 27 wherein said producing a fourth implementation further comprises:

receiving a reference name;

producing an implementation that calls a handler method of the superclass of said class, passing said reference name as an argument and returning the reference value associated with said reference name; and

returning said reference value.

33. The method of claim 27 wherein said producing a fifth implementation further comprises:

receiving an operation name and any associated arguments;

producing an implementation that calls a handler method of the superclass of said class, passing said operation name and said associated arguments as arguments and returning an operation return value; and

returning said operation return value.

34. A program storage device readable by a machine, embodying a program of instructions executable by the machine to perform a method to dynamically implement a Java™ Metadata Interface (JMI) to a metamodel, the method comprising:

receiving a JMI implementation request, said request associated with a metamodel, said metamodel comprising at least one package, said at least one package comprising at least one class, said at least one class comprising at least one attribute, reference or operation;

implementing a package proxy JMI interface when said request comprises a package proxy request;

implementing a class proxy JMI interface when said request comprises a class proxy request; and

implementing a class instance JMI interface when said request comprises a class instance request.

35. The program storage device of claim 34 wherein said implementing a package proxy JMI interface comprises:

generating bytecode for a class that implements said package proxy JMI interface;

creating a new instance of said class; and

returning said instance.

36. The program storage device of claim 35 wherein said generating further comprises:

receiving a metamodel package;

receiving a package proxy interface method associated with said metamodel package;

determining a class name based upon said interface method;

searching said metamodel package for a class corresponding to said class name; and

producing an implementation of said interface method that returns the proxy for said class when said class name is found in said package.

37. The program storage device of claim 36 wherein said implementation of said interface method calls a handler method of the superclass of said class, passing said class name as an argument and returning the proxy for said class.

38. The program storage device of claim 34 wherein said implementing a class proxy JMI interface comprises:

generating bytecode for a class that implements said class proxy JMI interface;

creating a new instance of said class; and

returning said instance.

39. The program storage device of claim 38 wherein said generating further comprises:

receiving a metamodel class;

receiving a class proxy interface method associated with said metamodel class;

producing a first implementation of said interface method that creates a new instance of said class when said method is parameterless; and

producing a second implementation of said interface method that creates a new instance of said class and sets the attributes passed as arguments to said interface method when said interface method includes at least one parameter.

40. The program storage device of claim 39 wherein said first implementation calls a handler method of the superclass of said class, passing said class name as an argument and returning a new instance of said class.

41. The program storage device of claim 39 wherein said second implementation calls a handler method of the superclass of said class, passing said class name, attributes and attribute values as arguments and returning a new instance of said class.

42. The program storage device of claim 34 wherein said implementing a class instance JMI interface comprises:

generating bytecode for a class that implements said class instance JMI interface;

creating a new instance of said class; and

returning said instance.

43. The program storage device of claim 42 wherein said generating further comprises:

receiving a metamodel class;

receiving a class instance interface method associated with said metamodel class, said interface method having an interface method name;

producing a first implementation of said interface method that sets the value of an attribute when said interface method name includes a first prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a second implementation of said interface method that sets the value of a reference when said interface method name includes a first prefix and when the reference associated with said interface method is found in said metamodel class;

producing a third implementation of said interface method that gets the value of an attribute when said interface method name includes a second prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a fourth implementation of said interface method that gets the value of a reference when said interface method name includes a second prefix and when the reference associated with said interface method is found in said metamodel class; and

producing a fifth implementation of said interface method that executes an operation when said interface method has the same name as said operation.

44. The program storage device of claim 43 wherein

said first prefix is "set"; and

said second prefix is "get".

45. The program storage device of claim 43 wherein said producing a first implementation further comprises:

receiving an attribute name and an attribute value; and

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name and attribute value as arguments.

46. The program storage device of claim 43 wherein said producing a second implementation further comprises:

receiving a reference name and an reference value; and

producing an implementation that calls a handler method of the superclass of said class, passing said reference name and reference value as arguments.

47. The program storage device of claim 43 wherein said producing a third implementation further comprises:

receiving an attribute name;

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name as an argument and returning the attribute value associated with said attribute name; and

returning said attribute value.

48. The program storage device of claim 43 wherein said producing a fourth implementation further comprises:

receiving a reference name;

producing an implementation that calls a handler method of the superclass of said class, passing said reference name as an argument and returning the reference value associated with said reference name; and

returning said reference value.

49. The program storage device of claim 43 wherein said producing a fifth implementation further comprises:

receiving an operation name and any associated arguments;

producing an implementation that calls a handler method of the superclass of said class, passing said operation name and said associated arguments as arguments and returning an operation return value; and

returning said operation return value.

50. A program storage device readable by a machine, embodying a program of instructions executable by the machine to perform a method to dynamically implement a Java™ Metadata Interface (JMI) to a metamodel, the method comprising:

receiving a JMI implementation request, said request associated with a metamodel, said metamodel comprising at least one package, said at least one package comprising at least one class, said at least one class comprising at least one attribute, reference or operation;

implementing a JMI interface when said JMI interface is unimplemented; and

executing a stored JMI interface implementation when said JMI interface is implemented.

51. The program storage device of claim 50 wherein

said implementing further comprises:

implementing a package proxy JMI interface when said request comprises a package proxy request and when said package proxy JMI interface is unimplemented;

implementing a class proxy JMI interface when said request comprises a class proxy request and when said class proxy JMI interface is unimplemented; and

implementing a class instance JMI interface when said request comprises a class instance request and when said class instance JMI interface is unimplemented; and

said executing further comprises:

executing a stored a package proxy JMI interface implementation when said request comprises a package proxy request and when said package proxy JMI interface is implemented;

executing a stored class proxy JMI interface when said request comprises a class proxy request and when said class proxy JMI interface is implemented; and

executing a stored class instance JMI interface when said request comprises a class instance request and when said class instance JMI interface is implemented.

52. The program storage device of claim 51 wherein said implementing a package proxy JMI interface comprises:

generating bytecode for a class that implements said package proxy JMI interface;

creating a new instance of said class; and

returning said instance.

53. The program storage device of claim 52 wherein said generating further comprises:

receiving a metamodel package;

receiving a package proxy interface method associated with said metamodel package;

determining a class name based upon said interface method;

searching said metamodel package for a class corresponding to said class name; and

producing an implementation of said interface method that returns the proxy for said class when said class name is found in said package.

54. The program storage device of claim 53 wherein said implementation of said interface method calls a handler method of the superclass of said class, passing said class name as an argument and returning the proxy for said class.

55. The program storage device of claim 51 wherein said implementing a class proxy JMI interface comprises:

generating bytecode for a class that implements said class proxy JMI interface;

creating a new instance of said class; and

returning said instance.

56. The program storage device of claim 55 wherein said generating further comprises:

receiving a metamodel class;

receiving a class proxy interface method associated with said metamodel class;

producing a first implementation of said interface method that creates a new instance of said class when said method is parameterless; and

producing a second implementation of said interface method that creates a new instance of said class and sets the attributes passed as arguments to said interface method when said interface method includes at least one parameter.

57. The program storage device of claim 56 wherein said first implementation calls a handler method of the superclass of said class, passing said class name as an argument and returning a new instance of said class.

58. The program storage device of claim 56 wherein said second implementation calls a handler method of the superclass of said class, passing said class name, attributes and attribute values as arguments and returning a new instance of said class.

59. The program storage device of claim 51 wherein said implementing a class instance JMI interface comprises:

generating bytecode for a class that implements said class instance JMI interface;

creating a new instance of said class; and

returning said instance.

60. The program storage device of claim 59 wherein said generating further comprises:

receiving a metamodel class;

receiving a class instance interface method associated with said metamodel class, said interface method having an interface method name;

producing a first implementation of said interface method that sets the value of an attribute when said interface method name includes a first prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a second implementation of said interface method that sets the value of a reference when said interface method name includes a first prefix and when the

reference associated with said interface method is found in said metamodel class;

producing a third implementation of said interface method that gets the value of an attribute when said interface method name includes a second prefix and when the attribute associated with said interface method is found in said metamodel class;

producing a fourth implementation of said interface method that gets the value of a reference when said interface method name includes a second prefix and when the reference associated with said interface method is found in said metamodel class; and

producing a fifth implementation of said interface method that executes an operation when said interface method has the same name as said operation.

61. The program storage device of claim 60 wherein

said first prefix is "set"; and

said second prefix is "get".

62. The program storage device of claim 60 wherein said producing a first implementation further comprises:

receiving an attribute name and an attribute value; and

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name and attribute value as arguments.

63. The program storage device of claim 60 wherein said producing a second implementation further comprises:

receiving a reference name and an reference value; and

producing an implementation that calls a handler method of the superclass of said class, passing said reference name and reference value as arguments.

64. The program storage device of claim 60 wherein said producing a third implementation further comprises:

receiving an attribute name;

producing an implementation that calls a handler method of the superclass of said class, passing said attribute name as an argument and returning the attribute value associated with said attribute name; and

returning said attribute value.

65. The program storage device of claim 60 wherein said producing a fourth implementation further comprises:

receiving a reference name;

producing an implementation that calls a handler method of the superclass of said class, passing said reference name as an argument and returning the reference value associated with said reference name; and

returning said reference value.

66. The program storage device of claim 60 wherein said producing a fifth implementation further comprises:

receiving an operation name and any associated arguments;

producing an implementation that calls a handler method of the superclass of said class, passing said operation name and said associated arguments as arguments and returning an operation return value; and

returning said operation return value.